SOFTWARE DE BORDO PARA UM CUBESAT (NANOSATC-BR)

RELATÓRIO FINAL DE PROJETO DE INICIAÇÃO CIENTÍFICA
PIBIC/INPE - CNPq/MCT

PROCESSO Nº: 102668/2009-4

Lucas Antunes Tambara – Bolsista PIBIC/INPE – CNPq/MCT
Laboratório de Informática do Projeto NanoSatC-BR
CRS/INPE – MCT
Centro Regional Sul de Pesquisas Espaciais
CRS/INPE - MCT
E-mail: tambara@lacesm.ufsm.br

Dr. Otavio Santos Cupertino Durão – Orientador
Coordenação de Planejamento Estratégico e Avaliação
CPA/DIR/INPE – MCT
Instituto Nacional de Pesquisas Espaciais
INPE - MCT
E-mail: durao@dir.inpe.br

Santa Maria, junho de 2009

RELATÓRIO FINAL DE INICIAÇÃO CIENTÍFICA DO PROGRAMA: PIBIC/INPE – CNPq/MCT

PROJETO

SOFTWARE DE BORDO PARA UM CUBESAT (NANOSATC-BR)

PROCESSO N°: 102668/2009-4

Relatório elaborado por:

Lucas Antunes Tambara – Bolsista PIBIC/INPE – CNPq/MCT
E-mail: tambara@lacesm.ufsm.br

Dr. Otavio Santos Cupertino Durão – Orientador
Coordenação de Planejamento Estratégico e Avaliação
CPA/DIR/INPE – MCT
E-mail: durao@dir.inpe.br

Dr. Nelson Jorge Schuch – Co-Orientador
Centro Regional Sul de Pesquisas Espaciais
CRS/INPE – MCT
E-mail: njschuch@lacesm.ufsm.br

DADOS DE IDENTIFICAÇÃO

Projeto:

SOFTWARE DE BORDO PARA UM CUBESAT (NANOSATC-BR)

Processo CNPq: Nº 102668/2009-4.

Bolsista:

Lucas Antunes Tambara.

Acadêmico de Ciência da Computação.

Centro de Tecnologia - Universidade Federal de Santa Maria – UFSM.

Orientador:

Dr. Otavio Santos Cupertino Durão.

Coordenação de Planejamento Estratégico e Avaliação – CPA/DIR/INPE – MCT.

Co-Orientador:

Dr. Nelson Jorge Schuch.

Centro Regional Sul de Pesquisas Espaciais – CRS/INPE – MCT.

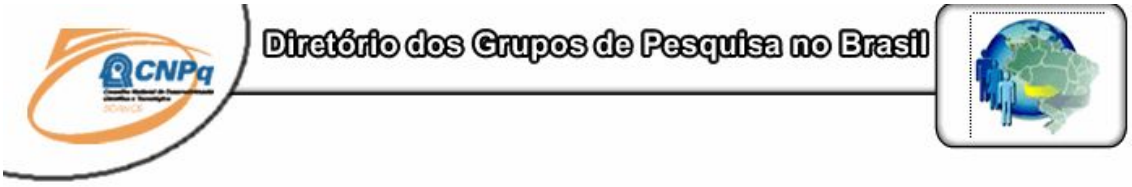
Colaboradores/ Acadêmicos:

Eng. Mestre Maria de Fátima Mattiello-Francisco – Coordenadora Geral do Convênio INPE/MCT – UFSM, por parte do INPE/MCT.

Lucas L. Costa – Aluno do Curso de Engenharia Mecânica da UFSM, Estagiário do CRS/INPE – MCT no Projeto NANOSATC-BR.

Local de Trabalho/Execução do Projeto:

Laboratório de Informática do Projeto NANOSATC-BR do
CRS/INPE – MCT, Santa Maria, RS.
Projeto executado no âmbito da Parceria INPE/MCT – UFSM através do Laboratório de Ciências Espaciais de Santa Maria – LACESM/CT-UFSM.



Grupo de Pesquisa
Clima Espacial, Magnetosferas, Geomagnetismo: Interações Terra - Sol, NanoSatC-Br

Identificação | Recursos Humanos | Linhas de Pesquisa | Indicadores do Grupo

Identificação

Dados básicos

Nome do grupo: Clima Espacial, Magnetosferas, Geomagnetismo: Interações Terra - Sol, NanoSatC-Br

Status do grupo: **certificado pela instituição**

Ano de formação: 1996

Data da última atualização: 11/06/2009 11:14

Líder(es) do grupo: Nelson Jorge Schuch - nelson.schuch@pq.cnpq.br
Natanael Rodrigues Gomes - natanael.gomes@lacesm.ufsm.br

Área predominante: Ciências Exatas e da Terra; Geociências

Instituição: Instituto Nacional de Pesquisas Espaciais - INPE

Órgão: Coordenação de Gestão Científica - CIE

Unidade: Centro Regional Sul de Pesquisas Espaciais - CRS

Endereço

Logradouro: Caixa Postal 5021

Bairro: Camobi

CEP: 97110970

Cidade: Santa Maria

UF: RS

Telefone: 33012026

Fax: 33012030

E-mail: njschuch@lacesm.ufsm.br

Home page: http://

Repercussões dos trabalhos do grupo

O Grupo - CLIMA ESPACIAL, MAGNETOSFERAS, GEOMAGNETISMO: INTERAÇÃO TERRA-SOL do Centro Regional Sul de Pesquisas Espaciais - CRS/INPE-MCT, em Santa Maria, e Observatório Espacial do Sul - OES/CRS/INPE - MCT, Lat. 29º26´24"S, Long. 53º48´38"W, Alt. 488m, em São Martinho da Serra, RS, criado por Nelson Jorge Schuch em 1996, colabora com pesquisadores da: UFSM (CT-LACESM), INPE, CRAAM-Universidade P. Mackenzie, IAG/USP, OV/ON, DPD/UNIVAP e SEFET/GO, no Brasil e internacionais do: Japão (Universidades: Shinshu, Nagoya, Kyushu, Takushoku e National Institute of Polar Research), EUA ((Bartol Research Institute/University of Delaware e NASA (Jet Propulsion Laboratory e Goddard Space Flight Center)), Alemanha (University of Greifswald e Max Planck Institute for Solar System Research), Australia (Australian Government Antarctic Division e University of Tasmania), Armênia (Alikhanyan Physics Institute) e Kuwait (Kuwait University). Linhas de Pesquisas: MEIO INTERPLANETÁRIO - CLIMA ESPACIAL, MAGNETOSFERAS x GEOMAGNETISMO, AERONOMIA - IONOSFERAS x AEROLUMINESCÊNCIA, NANOSATC-BR. Áreas de interesse: Heliosfera, Física Solar, Meio Interplanetário, Clima Espacial, Magnetosferas, Geomagnetismo, Aeronomia, Ionosferas, Aeroluminescência, Raios Cósmicos, Muons, Pequenos Satélites Científicos. Objetivos: Pesquisar o acoplamento energético na Heliosfera, mecanismos de geração de energia no Sol, Vento Solar, sua propagação no Meio Interplanetário, acoplamento com as magnetosferas planetárias, no Geoespaço com a Ionosfera e a Atmosfera Superior, previsão de ocorrência de tempestades magnéticas e das intensas correntes induzidas na superfície da Terra,Eletricidade Atmosferica e seus Eventos Luminosos Transientes (TLEs). As Pesquisas base de dados de sondas no Espaço Interplanetário e dentro de magnetosferas planetárias, e de modelos computacionais físicos e estatísticos.Vice-Líderes: Alisson Dal Lago, Nalin Babulau Trivedi, Otávio Santos Cupertino Durão, Natanael Rodrigues Gomes.

Recursos humanos

Pesquisadores	Total: 41

Ademar Michels	Jean Pierre Raulin
Alan Prestes	Joao Paulo Minussi
Alicia Luisa Clúa de Gonzalez	Jose Humberto Andrade Sobral
Alisson Dal Lago	Juliano Moro
Antonio Claret Palerosi	Mangalathayil Ali Abdu
Barclay Robert Clemesha	Marcelo Barcellos da Rosa
Caitano Luiz da Silva	Marcos Vinicius Dias Silveira
Carlos Roberto Braga	Nalin Babulal Trivedi
Clezio Marcos De Nardin	Natanael Rodrigues Gomes
Cristiano Max Wrasse	Nelson Jorge Schuch
Delano Gobbi	Nivaor Rodolfo Rigozo
Eurico Rodrigues de Paula	Odim Mendes Junior
Ezequiel Echer	Osmar Pinto Junior
Fabiano Luis de Sousa	Otavio Santos Cupertino Durão
Fábio Augusto Vargas dos Santos	Pawel Rozenfeld
Fernanda de São Sabbas Tavares	Petrônio Noronha de Souza
Fernando Luís Guarnieri	Polinaya Muralikrishna
Gelson Lauro Dal' Forno	Rajaram Purushottam Kane
Hisao Takahashi	Severino Luiz Guimaraes Dutra
Ijar Milagre da Fonseca	Walter Demetrio Gonzalez Alarcon
Jean Carlo Santos	

Estudantes	**Total: 30**
Aline Seeger Santos	Josemar de Siqueira
Bernardo Henz	Lilian Piecha Moor
Carlos Pinto da Silva Neto	Lucas Antunes Tambara
Cassio Espindola Antunes	Lucas Lopes Costa
Celito Muck Felipetto	Lucas Ramos Vieira
Claudio Machado Paulo	Luis Fernando Nicolini
Cristiano Sarzi Machado	Nikolas Kemmerich
Eduardo Escobar Bürger	Rafael Lopes Costa
Eduardo Weide Luiz	Ricardo Cartier dos Santos
Fernando de Souza Savian	Rodrigo da Rosa Azambuja
Guilherme Aluizio Steffens Lorenset	Rubens Zolar Gehlen Bohrer
Guilherme Grams	Tardelli Ronan Coelho Stekel
Guilherme Simon da Rosa	Thalis José Girardi
Igor Freitas Fagundes	Tiago Jaskulski
Jose Fernando Thuorst	Willian Rigon Silva

Técnicos	**Total: 2**

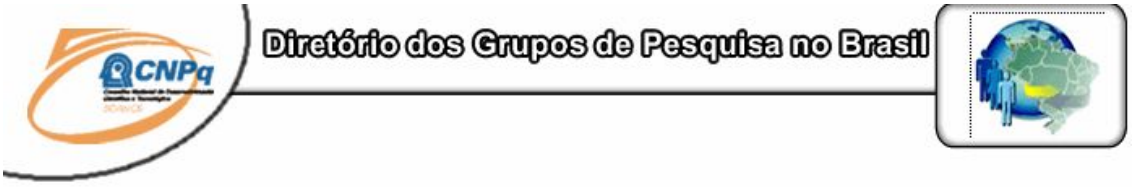
Eduardo Ceretta Dalla Favera - Ensino Profissional de nível técnico - Técnico em Computação
Vinicius Ceregati Costa - Graduação - \Outra Função

Linhas de pesquisa	**Total: 4**

- AERONOMIA - IONOSFERAS x AEROLUMINESCÊNCIA
- Desenvolvimento de CubeSats - NANOSATC-BR
- MAGNETOSFERAS x GEOMAGNETISMO
- MEIO INTERPLANETÁRIO - CLIMA ESPACIAL

Relações com o setor produtivo	**Total: 0**

Indicadores de recursos humanos do grupo	
Integrantes do grupo	**Total**
Pesquisador(es)	41
Estudante(s)	30
Técnico(s)	2



Linha de Pesquisa
Desenvolvimento de CubeSats - NANOSATC-BR

Linha de pesquisa

Desenvolvimento de CubeSats - NANOSATC-BR

Nome do grupo: Clima Espacial, Magnetosferas, Geomagnetismo: Interações Terra - Sol, NanoSatC-Br

Palavras-chave: CubeSats; Desenvolvimento de Engenharias - Tecnologias; Miniaturização; Nanosatélites; Nanotecnologia; Pesquisa do Geoespaço;

Pesquisadores:

Ademar Michels
Alicia Luisa Clúa de Gonzalez
Alisson Dal Lago
Antonio Claret Palerosi
Clezio Marcos De Nardin
Ezequiel Echer
Fabiano Luis de Sousa
Fernando Luís Guarnieri
Ijar Milagre da Fonseca
Jean Pierre Raulin
Jose Humberto Andrade Sobral
Nalin Babulal Trivedi
Natanael Rodrigues Gomes
Nelson Jorge Schuch
Nivaor Rodolfo Rigozo
Odim Mendes Junior
Otavio Santos Cupertino Durão
Pawel Rozenfeld
Petrônio Noronha de Souza
Severino Luiz Guimaraes Dutra
Walter Demetrio Gonzalez Alarcon

Estudantes:

Bernardo Henz
Cassio Espindola Antunes
Celito Muck Felipetto
Eduardo Escobar Bürger
Fernando de Souza Savian
Guilherme Grams
Guilherme Simon da Rosa
Igor Freitas Fagundes
Jose Fernando Thuorst
Josemar de Siqueira
Lucas Antunes Tambara
Lucas Lopes Costa
Lucas Ramos Vieira
Luis Fernando Nicolini
Nikolas Kemmerich
Rafael Lopes Costa
Ricardo Cartier dos Santos
Rubens Zolar Gehlen Bohrer
Tardelli Ronan Coelho Stekel
Tiago Jaskulski
Willian Rigon Silva

Árvore do conhecimento:

Ciências Exatas e da Terra; Astronomia; Astrofísica do Sistema Solar;
Ciências Exatas e da Terra; Geociências; Instrumentação Científica;

Engenharias; Engenharia Aeroespacial; Engenharia Aeroespacial - Pequenos Satélites;

Setores de aplicação:

Aeronáutica e espaço

Objetivo:

Pesquisas: Geoespaço e em Engenharias/Tecnologias: eletrônica, comunicações, mecânica, lançamento de pequenos satélites científico universitário - iniciação científica: CubeSat (100g-1Kg, 10x10x10cm), Nanosatélite (1Kg-10Kg); Carga útil: magnetômetro e detector de partículas; Desenvolvimentos: estrutura mecânica, computador-bordo, programas, estação terrena, testes/integração, sub-sistemas: potencia, propulsão, telemetria, controle: atitude, térmico, Vice-Líder: Otávio Santos Cupertino Durão

AGRADECIMENTOS

Agradeço meu orientador, Dr. Otavio Santos Cupertino Durão e meu Co-Orientador e mentor Dr. Nelson Jorge Schuch pela atenção e apoio prestados em todas as dificuldades encontradas no decorrer do trabalho desenvolvido, gerando grande crescimento pessoal.

Meus sinceros agradecimentos: (i) aos funcionários, servidores do CRS/INPE – MCT e do LACESM/CT – UFSM pelo apoio e pela infra-estrutura disponibilizada; (ii) ao Programa PIBIC/INPE – CNPq/MCT pela aprovação do Projeto de Pesquisa, que me permitiu dar os primeiros passos na iniciação científica e tecnológica, propiciando grande crescimento profissional; (iii) ao Coordenador Dr. José Carlos Becceneri e a Secretária do Programa PIBIC/INPE – CNPq/MCT, Sra. Egidia Inácio da Rosa, pelo constante apoio, alertas e sua incansável preocupação com toda a burocracia e datas limites do Programa para com os bolsistas de I. C. & T. do CRS/INPE - MCT.

SUMÁRIO

RESUMO

O relatório apresenta as atividades de pesquisa vinculadas ao Programa PIBIC/INPE – CNPq/MCT realizadas pelo bolsista **Lucas Antunes Tambara**, durante o período de abril de 2009 a julho de 2009, no Projeto "**SOFTWARE DE BORDO PARA UM CUBESAT (NANOSATC-BR)**" junto ao Centro Regional Sul de Pesquisas Espaciais – CRS/INPE-MCT. As atividades foram desenvolvidas no **Laboratório de Informática do Projeto NanoSatC-BR do CRS/INPE-MCT** no âmbito da Parceria: INPE/MCT – UFSM, através do Laboratório de Ciências Espaciais se Santa Maria – LACESM/CT – UFSM.

O trabalho tem por objetivo estudar os requisitos para o desenvolvimento de um aplicativo de bordo para um satélite da classe dos CubeSats compatível com a Missão NanoSatC-BR. O estudo foi realizado analisando-se a arquitetura do computador de bordo do satélite e o fluxo de dados interno do computador. Por fim, é apresentada a estruturação de um aplicativo de bordo compatível com a Missão NanoSatC-BR.

CAPÍTULO 1

1.1. INTRODUÇÃO

O Relatório é composto por descrições das atividades de pesquisa na área espacial referentes a pequenos satélites e voltado especificamente para uma classe de nanosatélites, os CubeSats, com identificação de conceitos, aplicações, estrutura, funcionamento e seu projeto. O relatório tem ênfase no Projeto de um Aplicativo de Bordo para um CubeSat e assuntos relacionados, que englobam conhecimentos de ciências básicas, tecnologia espacial, funcionamento de todo satélite, ferramentas de projeto e possíveis soluções aplicáveis para satélites dessa classe.

É realizada a análise de aplicativos de bordo espaciais com a finalidade de verificar a viabilidade de uso das soluções no Projeto do Nanosatélite Científico Acadêmico Brasileiro – NanoSatC-BR.

A divisão dos capítulos representa a evolução da Pesquisa que, inicialmente, teve foco na familiarização de conceitos relacionados à aplicativos de bordo em satélites de uma forma geral, para posterior entendimento dos conceitos referentes a classe dos CubeSats. Ainda, pesquisas de conhecimentos básicos de tecnologia espacial, fluxo de dados e engenharia de software também são incluídos.

1.2. OBJETIVO DO PROJETO

O Projeto tem por objetivo principal a obtenção de conhecimento de conceitos de forma suficiente para viabilizar a estruturação de um Aplicativo de Bordo para um CubeSat, com identificação de requisitos, plataforma de desenvolvimento disponível e, ainda, a aplicação direta no Projeto NanoSatC-

BR, onde o bolsista é o aluno responsável pelo subsistema de Computador de Bordo.

O fomento da pesquisa na área espacial, muito pouco explorada no Brasil, bem como a preocupação com o desenvolvimento que esta área pode trazer para a tecnologia e a formação de recursos humanos é outro objetivo a ser considerado. Ainda, ressalta-se que a área espacial traz grandes satisfações ao bolsista, representando forte atrativo para seu desenvolvimento profissional.

1.3. METODOLOGIA

O trabalho foi desenvolvido através de extensa revisão bibliográfica de assuntos básicos sobre satélites, Subsistemas de Computador de Bordo e todo contexto envolvido em missões espaciais, para posterior aplicação e entendimento da classe dos CubeSats.

Através de pesquisa exploratória (Internet, livros, artigos científicos e contato com profissionais diretamente ligados a Projeto de Computadores de Bordo de satélites, foram estudados aplicativos de bordo para esta classe de satélites.

Ainda fizeram parte da pesquisa, conceitos de sistemas operacionais e sistemas de tempo real, através dos quais uma estrutura de aplicativo foi obtida levantando a possibilidade de sua utilização no Projeto do Aplicativo de Bordo do NanoSatC-Br.

CAPÍTULO 2

2.1. NANOSATC-BR

A Missão NanoSatC-BR – Clima Espacial consiste em um Programa Integrado de Pesquisa Espacial com desenvolvimento de Engenharias e Tecnologias Espaciais através de um pequeno satélite, o Nanosatélite Científico Brasileiro, NanoSatC-BR.

Seu objetivo é científico e sócio-educativo, tendendo à formação de Recursos Humanos especializados, além de realizar monitoramento, em tempo real, no âmbito do clima espacial, o geoespaço e os distúrbios observados na magnetosfera terrestre – campo geomagnético e a precipitação de partículas energéticas, sobre o território brasileiro, determinando seus efeitos na região da Anomalia Magnética do Atlântico Sul – AMAS e do Eletrojato da Ionosfera Equatorial.

A Missão consiste na adaptação de magnetômetros, detectores de partículas e, simultaneamente, da montagem, qualificação e lançamento de um satélite científico Brasileiro, o NanoSatC-BR, miniaturizado do tipo CubeSat.

O lançamento do NanoSatC-BR será em órbita baixa polar do tipo Sol-Síncrona, em torno de 650 km, o satélite levará como carga útil, em princípio, dois experimentos, um magnetômetro (para medidas do campo magnético terrestre) e um detector de partículas (para medição da quantidade de partículas precipitadas próximo à superfície terrestre), adaptados e integrados por estudantes universitários participantes do Projeto.

O subsistema de computador de bordo, apresentado neste relatório com maiores detalhes, o subsistema de potência no qual estão definidos os painéis solares que poderão fornecer até 2 W por face, o subsistema de controle térmico passivo, entre outros subsistemas e seus componentes estão em desenvolvimento por alunos de diferentes áreas do conhecimento com suporte de cientistas, engenheiros e tecnólogos especialistas nas respectivas áreas de

atuação. O lançamento está previsto para o final de 2010 com lançador ainda não definido, provavelmente indiano. Destaca-se que o Projeto do Aplicativo de Bordo para um CubeSat (NanoSatC-BR) está sendo desenvolvido como atividade de pesquisa aplicada do bolsista.

O satélite pertence à classe de nanosatélites, os CubeSats, satélites em formato cúbico de dimensões 100x100x100 mm, de forma que os subsistemas devem ser desenvolvidos respeitando restrições de volume, massa e consumo de energia, necessitando-se do uso de componentes miniaturizados.

O Projeto é uma iniciativa do Centro Regional Sul de Pesquisas Espaciais – CRS/INPE – MCT em parceria com a UFSM e outras instituições brasileiras e está sendo desenvolvido com a participação direta de estudantes, tornando-se um dos primeiros satélites universitários desenvolvidos no Brasil.

Figura 01: CP-1, o primeiro nanosatélite da Universidade Politécnica da Califórnia (Cal Poly).[1]

2.2. CUBESATS

[1] Disponível em <http://en.wikipedia.org/wiki/File:Cp1.jpg>. Acesso em: 03/04/2009.

O CubeSat é um satélite da classe dos nanosatélites e tem como principais características o seu formato cúbico com 10 cm de aresta (100x100x100mm) e massa limite de até 1 kg.

Estes satélites têm maior utilização em missões científicas para obtenção de dados, testes para dispositivos, materiais inéditos e até a obtenção de imagens.

Sua utilização se deve principalmente ao fato de que a divisão de tarefas em satélites menores acarreta menores custos e, caso ocorra a perda destes, não serão orçamentos exorbitantes perdidos. Além de ser uma ótima ferramenta educativa para o envolvimento de alunos de graduação, os quais têm oportunidade de desenvolver um projeto real de satélite.

As restrições de massa e volume interno (aproximadamente 1litro) levam a prazos e custos de desenvolvimento bastante reduzidos, o caso do NanoSatC-BR, por exemplo, possui a previsão de apenas dois anos para o seu desenvolvimento. Ainda, o custo de lançamento é uma das maiores vantagens, pois, devido a sua pequena massa, esta na faixa de milhares de dólares, representando enorme vantagem se comparado com os milhões gastos em satélites maiores.

Além da opção de lançamento de carona com um satélite principal, geralmente de grande porte, e outros satélites pequenos, há também a possibilidade de um lançamento exclusivamente de CubeSats. A interface criada entre o CubeSat e o lançador foi um grande facilitador e o fator que viabilizou o desenvolvimento destes e redução de preços, o P-POD foi um dispositivo criado para facilitar o lançamento através de um mecanismo simples de ejeção do CubeSat em órbita.

Este tipo de satélite vem sendo desenvolvido por universidades de vários países devido, principalmente, ao reduzido custo de produção e por ser uma excelente oportunidade para alunos universitários de várias áreas das

ciências e tecnologias aplicarem seus conhecimentos e o ter envolvimento em um projeto real de aplicação espacial.

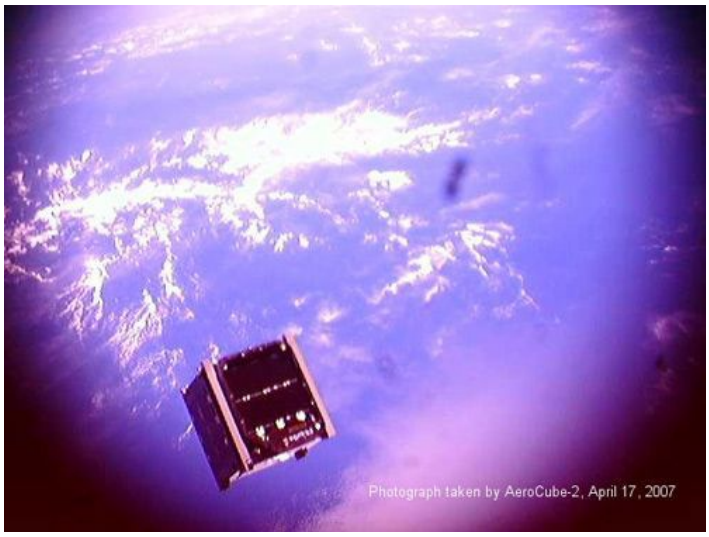


Figura 02: Foto de um CubeSat captada por outro CubeSat.[2]

A plataforma dos satélites artificiais é dividida em subsistemas. Isto é feito para sistematizar o trabalho de engenharia requerido no projeto, montagem e testes, dividindo-o em áreas de competência. Os subsistemas usualmente encontrados são os seguintes:

- Controle de Atitude (*Attitude Determination and Control* ou *Attitude Control System*).

  Objetivo: controlar a orientação do satélite no espaço.

  Partes: rodas de reação ou volantes de inércia, bobinas magnéticas, sensores de Sol, de Terra, de estrelas, magnetômetros e giroscópios.
- Suprimento de Energia (*Electrical Power and Distribution*).

  Objetivo: aquisição, armazenamento e fornecimento de energia necessária aos diversos subsistemas.

[2] Disponível em < http://www.space.com/>. Acesso em: 03/04/2009.

Partes: painéis solares e seus diversos acessórios, conversores e baterias.

- Telecomunicação de Serviço (*Telemetry, Tracking and Command*).

  Objetivo: enviar e receber os dados que permitem o acompanhamento do funcionamento e o comando do satélite. Há também os dados científicos provenientes das cargas úteis do satélite.

  Partes: transmissores, receptores e antenas.

- Gestão de Bordo (*Command and Data Handling*).

  Objetivo: processar as informações recebidas da ou a serem enviadas à Terra e as informações internas ao satélite.

  Partes: computador de bordo e seu aplicativo.

- Estrutura e Mecanismos (*Structures and Mechanisms*).

  Objetivo: fornecer suporte mecânico e de movimento para as partes do satélite. Também oferecer proteção contra as vibrações de lançamento e contra a radiação em órbita.

  Partes: estruturas primárias e secundárias, mecanismos de abertura de painéis solares e de separação do lançador, mecanismos de abertura de antenas, dispositivos pirotécnicos, mecanismos de extensão, alinhamento e suspensões com amortecedores.

- Controle Térmico (*Thermal Control*).

  Objetivo: manter os equipamentos dentro de suas faixas nominais de temperatura.

  Partes: aquecedores, isoladores, pinturas e radiadores.

- Propulsão (*Propulsion*).

  Objetivo: fornece o empuxo necessário para o controle da atitude e da órbita.

  Partes: bocais ou tubeiras, válvulas, reservatórios e tubulações.

Essa divisão de subsistemas também é aplicada para CubeSats, mas seus subsistemas estão integrados em um único módulo. Nesses, dependendo da sua missão, não é necessário uma grande precisão ou apontamento

específico, portando, não havendo necessidade de um subsistema de propulsão, assim como de um subsistema de controle de atitude muito preciso.

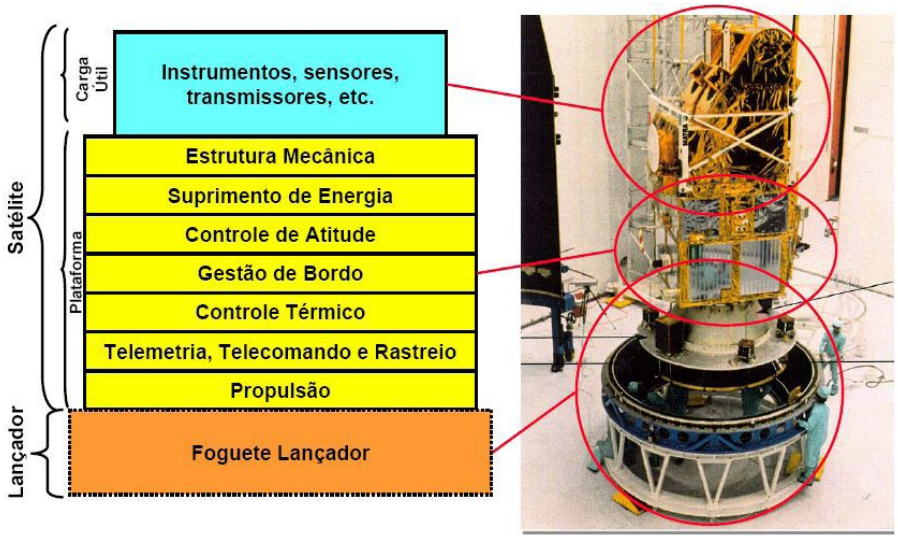


Figura 03: Partes de um satélite.[3]

[3] Fonte: DE SOUZA, P. N. Curso Introdutório de Tecnologia de Satélites. Instituto Nacional de Pesquisas Espaciais – INPE. São José dos Campos – SP, 2007.

CAPÍTULO 3

3.1. SISTEMAS DE TEMPO REAL

Sistemas de tempo real devem realizar uma quantidade específica de processamento dentro de um intervalo de tempo definido, tais como: controlar dispositivos externos, responder a eventos externos e compartilhar tempo de processamento entre múltiplos processos. Os sistemas de tempo real geram ações em resposta a eventos externos. Para isso, operações de controle e aquisição de dados são realizadas sob restrições de tempo e confiabilidade.

Um sistema de tempo real é formado por vários processos computacionais ou tarefas concorrentes. A concorrência decorre do fato de existirem tarefas assíncronas executadas em velocidades diferentes. Entretanto, de tempos em tempos, algumas tarefas precisam se comunicar e sincronizarem-se umas com as outras. Nestes sistemas, as entradas chegam de forma intermitente e não estão disponíveis de início, enquanto as saídas são geradas em resposta às entradas.

O termo sistema de tempo real normalmente refere-se ao sistema inteiro, que compreende a aplicação de tempo real, sistema operacional e subsistemas de entrada e saída, que contêm dispositivos especiais para realizar interface com diversos sensores e atuadores.

Portanto, o aplicativo desenvolvido para um sistema de tempo real é classificado como aplicativo de tempo real e deve responder, monitorar, analisar e controlar elementos do mundo real de acordo com o tempo especificado no domínio do problema. Os elementos incluem um componente de coleta de dados que obtém e formata as informações provenientes de um ambiente externo, um componente de análise que transforma as informações conforme a aplicação é exigida, um componente de controle e saída que responde ao ambiente externo e um componente de monitoração que coordena

todos os demais componentes de forma que a resposta em tempo real possa ser mantida.

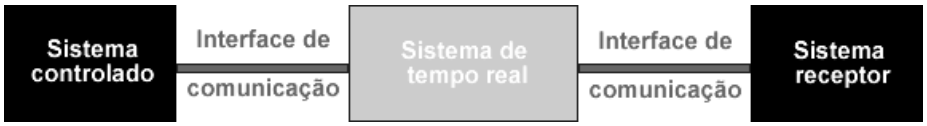


Figura 04: Interface de um sistema de tempo real.

Um método alternativo e recentemente explorado na área espacial é estruturar este tipo de aplicativo fazendo uso de um *framework*, isto é, utilizar uma abstração que une códigos comuns entre vários projetos de aplicativo provendo uma funcionalidade genérica. Um *framework* pode atingir uma funcionalidade específica, por configuração, durante a programação de uma aplicação.

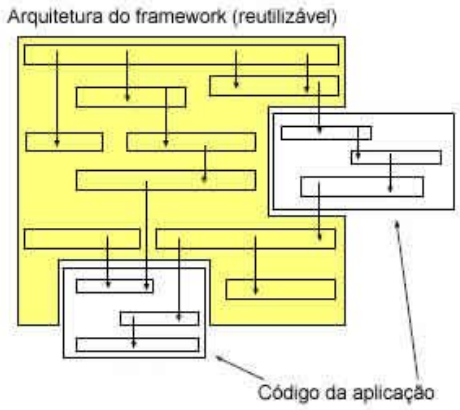


Figura 05: Desenvolvimento de um aplicativo baseado no uso de um *framework.*

3.2. APLICATIVOS DE BORDO EM SATÉLITES

Normalmente chamado de OBDH – *On Board Data Handling*, um aplicativo de bordo basicamente implementa as funções de “comando e controle” e “telemetria” a bordo de equipamentos espaciais. Esse sistema computacional embarcado tem por objetivo principal monitorar o comportamento e estado funcional dos demais subsistemas do satélite e

efetuar seus controles quando necessário, assim como garantir o armazenamento temporário e a integridade dos dados da missão.

Em geral, sistemas espaciais embarcados requerem controle em tempo real com alta confiabilidade e, por esta razão, os sistemas computacionais a bordo de satélites devem realizar serviços compatíveis com os demais elementos do veículo espacial e do segmento solo.

As funções de “comando e controle” a bordo de satélites possibilitam a operação remota dos seus subsistemas, tanto os que compõem a plataforma quanto os da carga útil ao longo da vida do satélite. Eles permitem que o controlador de solo corrija uma falha e opere um mecanismo conforme previsto, por exemplo. Estas operações devem ser realizadas de forma altamente confiável via telemetria. A facilidade de telecomando temporizado a bordo é importante para o controle do veículo e das cargas úteis ao longo de toda a órbita. Ela está fundamentada na capacidade do OBDH de interpretação dos códigos de telecomandos e consequente execução num instante de tempo estipulado por solo. O recurso de temporização a bordo possibilita que telecomandos enviados do solo, quando recebidos pelo satélite em visada pela estação solo, sejam executados posteriormente, fora de visada.

As funções associadas à “telemetria” a bordo da aeronave contemplam a monitoração em tempo real dos equipamentos e subsistemas da plataforma e também da carga útil em termos dos parâmetros críticos à sobrevivência da missão. Também são funções de telemetria no OBDH o empacotamento e armazenamento temporário dos dados da missão que serão transmitidos para solo. Entende-se por dados da missão as “telemetrias” da plataforma (parâmetros críticos e não críticos) e da carga útil. A telemetria da plataforma geralmente engloba os dados de “*housekeeping*”, ou seja, parâmetros de engenharia que precisam ser monitorados em solo para checar a saúde e o estado de operação dos equipamentos a bordo. A telemetria da carga útil consiste no conjunto de dados da aplicação, isto é nos dados científicos da missão.

A computação realizada pelo aplicativo de bordo em satélites também é utilizada para reduzir a telemetria das missões, aumentar a confiabilidade e contribuir para a redução de custos em outras partes da missão. Por exemplo, compressão de dados a bordo pode reduzir a taxa de bits de telemetria.

Muitas vezes, além do aplicativo de bordo embarcado no computador de bordo, há também um responsável pelo controle de atitude e órbita do satélite. Quando o aplicativo de controle de atitude compartilha o mesmo computador de bordo que o aplicativo de bordo ele é geralmente chamado de ACDH - *Attitude Board Data Handling*, ou Controle de Atitude e Supervisão de Bordo. Por outro lado, quando o aplicativo de controle de atitude possui um computador de bordo independente, ele é normalmente chamado de AOCS – *Attitude and Orbit Control Subsystem*, ou Subsistema de Controle de Atitude e Órbita. Esse sistema consiste de três elementos: sensores para medir erros de atitude, computador e aplicativo para calcular ações corretivas e atuadores para efetivar as correções. Com isso, o objetivo desse tipo de sistema é estabilizar o satélite através da comparação dos seus eixos de referência com outras fontes de referências disponíveis e conhecidas, que podem ser: as estrelas, o Sol, a Terra e o campo magnético terrestre.

3.3. FUNÇÕES DOS APLICATIVOS DE BORDO EM SATÉLITES

O aplicativo embarcado em um satélite está dividido hierarquicamente em sistema operacional e aplicativos. Com base no que foi descrito no item 3.2, pode-se enumerar as características destes dois aplicativos.

As principais características dos sistemas operacionais para satélites são:

- Resposta em tempo real, na qual a base de tempo está associada à aplicação;
- Escalonamento de processos;
- Sincronização e comunicação entre processos;

- Gerenciamento de memória;
- Gerenciamento dos dispositivos de entrada e saída;
- Tratamento e relato de erros;
- Implementação de mecanismos de tolerância a falhas por aplicativo.

Os aplicativos típicos dos sistemas a bordo de satélites realizam as seguintes funções:

- Comunicação de dados a bordo;
- Gerenciamento dos modos de operação de falhas;
- Interpretação e execução de telecomandos;
- Aquisição e processamento de telemetria;
- Processamento de dados a bordo;
- Controle e armazenamento de dados;
- Compartilhamento das funções do computador do sistema de controle de atitude.

De uma forma geral, as características mais relevantes do aplicativo embarcado são:

- Complexidade por ter interações com um ambiente complexo;
- Dedicação a uma aplicação específica, exigindo que o desenvolvimento siga padrões rigorosos;
- Utilização eficiente da capacidade de recursos do hardware, como processamento, memória e entrada e saída;
- Requisitos de análise e projetos especiais para seu desenvolvimento, que considerem seu efeito sobre a segurança do sistema em que está embutido.

CAPÍTULO 4

4.1. HARDWARE DO NANOSATC-BR

Para identificar os requisitos computacionais da missão e analisar os problemas que o aplicativo de bordo do NanoSatC-BR tem que solucionar, os componentes de hardware do NanoSatC-BR serão descritos a seguir. Uma maior ênfase é dada ao subsistema de computador de bordo pelo fato deste abrigar o aplicativo de bordo e portando, requerer um maior detalhamento de suas funcionalidades e periféricos.

4.1.1. Computador de bordo

Como em grande parte dos pequenos satélites, o NanoSatC-BR possui apenas um computador de bordo que é responsável pelas tarefas de comunicação entre os subsistemas, manipulação e armazenamento dos dados e por gerenciar a distribuição de energia.

O computador de bordo faz uso do módulo de vôo FM430, desenvolvido pela Pumpkin Inc. Este módulo de vôo conta com um microcontrolador da família MSP430, desenvolvido pela Texas Instruments. Algumas das principais características deste microcontrolador são descritas abaixo:

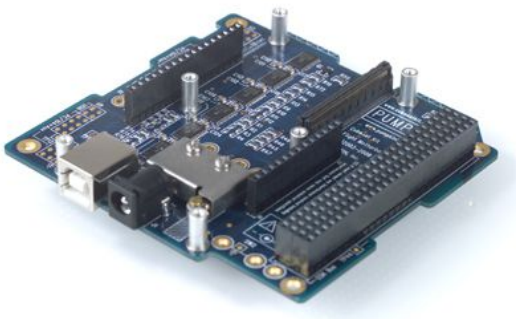


Figura 06: Módulo de vôo FM430 desenvolvido pela Pumpkin Inc.[4]

- Baixo consumo de energia: os microcontroladores da família MSP430 são geralmente conhecidos pelo baixo consumo de energia. Durante seu funcionamento normal, o consumo é da ordem de 250 µA/MIPS, para o armazenamento de dados em memória é gasto cerca de 0,1 µA e para o funcionamento no modo de relógio de tempo real é gasto em torno de 0,8 µA.
- Baixa tensão de operação: os MSP430 podem operar com tensões a partir de 1.8 V até 3.6 V.
- Alto desempenho: utilizam um barramento de dados de 16 bits, ao contrário da maioria dos outros microcontroladores de sua categoria, que utilizam um barramento de dados de 8 bits.
- Conjunto de instruções ortogonais: através da disponibilidade de qualquer modo de endereçamento para qualquer instrução e qualquer operando, é possível gerar códigos pequenos e eficientes, o que otimiza o desempenho dos compiladores de linguagens de alto nível, como C e C++.
- Número reduzido de instruções: a arquitetura RISC conta com apenas 27 instruções físicas e 24 emuladas, que são variações das instruções físicas, resultando em 51 instruções.
- Grande quantidade de periféricos: os MSP430 possuem um conjunto bastante grande de periféricos internos, como conversores analógico-digital de até 16 bits, conversores digital-analógico, comparador analógico, amplificador operacional programável, controladores de DMA (acesso direto à memória), temporizadores com diversos modos de funcionamento, controlador de LCD (monitor de cristal líquido), USARTs (padrão de comunicação de dados de forma serial) com capacidade de

[4] Disponível em <http://www.cubesatkit.com/content/datasheet.html>. Acesso em: 04/05/2009.

endereçamento, multiplicador por hardware com capacidade de executar operações de multiplicação e acúmulo, entre outros.

- Facilidade de gravação de dados e depuração de código: a utilização da interface JTAG (do acrônimo inglês *Joint Test Action Group*) para gravação de dados e depuração de código permite a programação e depuração de um aplicativo diretamente na placa de aplicação.
- Diversos encapsulamentos: suporta encapsulamentos desde 24 pinos até 100 pinos.

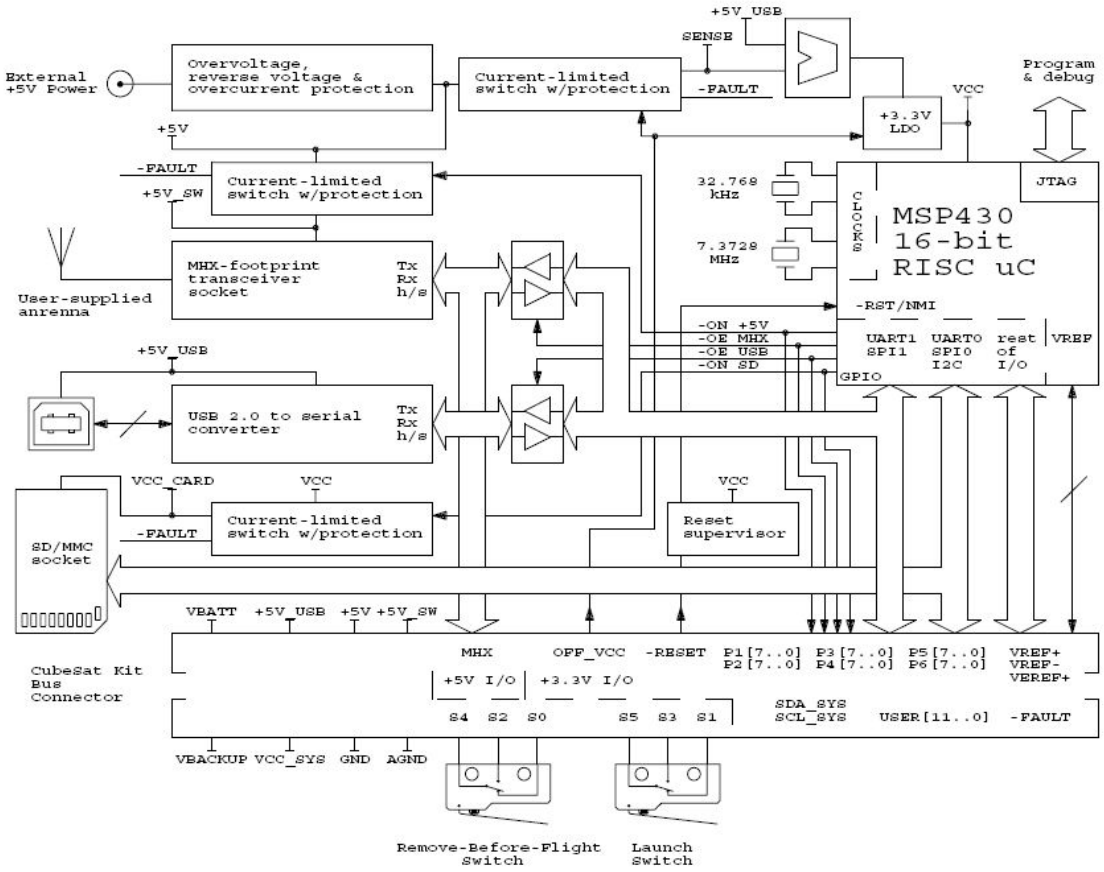


Figura 07: Diagrama de blocos do módulo de vôo FM430.[5]

Os microcontroladores MSP430 possuem uma estrutura simples e poderosa. Sua arquitetura RISC (*Reduced Instruction Set Computer* ou

[5] Disponível em <http://www.cubesatkit.com/docs/datasheet/DS_CSK_FM430_710-00252-C.pdf>. Acesso em 10/05/2009.

Computador com um Conjunto Reduzido de Instruções) combina um conjunto reduzido de instruções com uma arquitetura de barramento Von Neumann, permitindo que o microcontrolador possua um espaço único de endereçamento de memória. Desta forma, em tese, não há distinção entre memória de programa e memória de dados. Porém, funcionalmente, esta afirmação não é verdadeira, já que alguns endereços são reservados por registradores de acesso a periféricos, outros são utilizados para memória RAM de uso geral e há também endereços que são preenchidos com memórias FLASH.

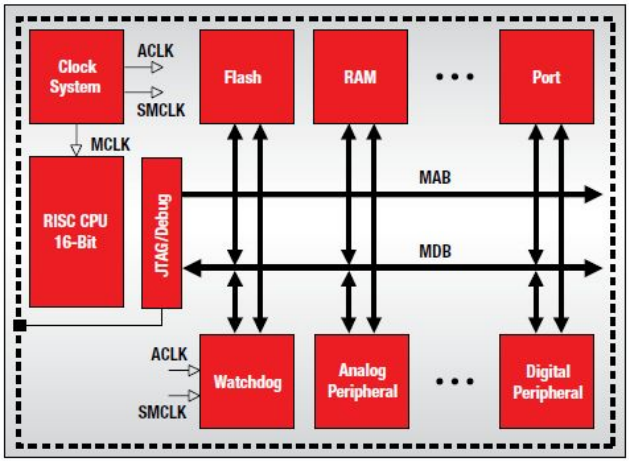


Figura 08: Aplicação da arquitetura de Von-Neumann ao MSP430.[6]

Esta família de microcontroladores possui três barramentos distintos, de endereços, de dados e de controle, sendo os dois primeiros de 16 bits. Uma vez que o barramento de endereços possui uma largura de 16 bits, significa que é possível acessar até 65536 posições de memória. Sendo o barramento de dados de 16 bits, é possível realizar operações envolvendo operandos de 8 ou 16 bits. Em uma operação de escrita de 8 bits, tendo um registrador do microcontrolador como destino, o byte mais significativo do registrador é preenchido com o valor zero, ao passo que em uma operação de leitura de 8 bits, tendo um registrador do microcontrolador como fonte, somente os 8 bits menos significativos são considerados.

[6] Disponível em <http://www.ti.com/msp430>. Acesso em 11/05/2009.

O sistema de *clock* dos MSP430, que permite a operação do microcontrolador e dos periféricos a partir de diferentes fontes, é chamado de BCS (*Basic Clock System* ou Sistema Básico de Clock) e é composto de um ou dois osciladores capazes de funcionar com cristais ou ressonantes externos, além de um oscilador interno controlado digitalmente. Os sinais provenientes desses osciladores podem ser selecionados para a geração de três sinais de *clock* do sistema: o sinal de *clock* principal, que é basicamente utilizado para sincronizar o microcontrolador e eventualmente outros periféricos do sistema; o sinal de *clock* secundário, que é utilizado como uma fonte de *clock* alternativa para diversos periféricos do microcontrolador; e o sinal de *clock* auxiliar, que possui a finalidade básica de funcionar como fonte de *clock* de precisão para periféricos durante modo de baixo consumo de energia.

O sistema de interrupções dos MSP430 é fixo e determinado. A latência de interrupção, isto é, o tempo decorrido entre o início do evento e o início da execução, é da ordem de seis ciclos de *clock* para que o microcontrolador reconheça a interrupção e efetue todo o procedimento de desvio do fluxo do programa.

Os MSP430 podem ter até seis portas de entrada ou saída de uso geral, cada uma com até oito pinos, totalizando 48 pinos de entrada ou saída. Cada um desses pinos pode ser configurado individualmente para funcionar como uma entrada ou uma saída.

O temporizador básico do MSP430 é formado por um contador de 8 bits. Ele pode ser utilizado como um divisor de freqüências programável e com capacidade de gerar interrupções. A sua principal aplicação é a geração de interrupções periódicas no microcontrolador, mas também pode funcionar como base de um sistema de relógio de tempo real.

A interface de comunicação de dados usada nos MSP430 é a USART (*Universal Synchronous Asynchronous Receiver Transmitter* ou Transmissor Receptor Universal Síncrono e Assíncrono). A USART implementada nos MSP430 suporta tamanhos de caracteres de 7 ou 8 bits, geração e detecção

automática de paridade e seleção dos números de bits de parada. Também estão implementados circuitos para detecção de erros de comunicação e para o suporte a endereçamento em sistemas multiprocessados. Este suporte a endereçamento pode ser implementado utilizando dois protocolos diferentes: detecção de linha inativa ou utilizando bits de endereçamento. A USART também possui um circuito gerador de *clock* que permite a obtenção de diversas velocidades de comunicação.

O protocolo utilizado na interface de comunicação dos MSP430 é o I²C (*Inter Integrated Comunication* ou Circuito Inter-Integrado). Ele é um protocolo síncrono *half-duplex* de duas linhas, do tipo mestre-escravo, ou seja, há um dispositivo que controla o fluxo de dados no barramento, pois somente um periférico pode estar transmitindo dados por vez. As duas linhas utilizadas são a de *clock* e outra de dados. Graças à especificação de saídas em dreno aberto, o protocolo permite a ligação de diversos dispositivos nas mesmas linhas, formando um autêntico barramento de comunicação serial, ou uma rede de dispositivos.

O conversor analógico-digital dos MSP430 atua sem intervenção do microcontrolador e com uma velocidade de até 200.000 amostras por segundo. Por não haver intervenção, este periférico conta com um acumulador de até 16 palavras, o que diminui o número de interrupções necessárias para leitura dos dados convertidos.

Assim como o conversor analógico-digital, há também um circuito com a finalidade de realizar cálculos matemáticos, mais especificamente multiplicações, que atua sem intervenção do microcontrolador. Este circuito é capaz de realizar multiplicações inteiras, sinalizadas ou não, envolvendo 8 e/ou 16 bits e contando com um acumulador de 32 bits. As operações que envolvem operandos sinalizados devem estar no formato de complemento de dois. Cada operação demanda de três ciclos de *clock* para ser concluída. Este circuito não realiza operações de ponto flutuante, sendo necessário haver uma implementação em nível de aplicativo para este tipo de cálculo.

Para automatizar operações de transferência de informação dentro do espaço de endereçamento da memória, alguns modelos de MSP430 contam com um controlador DMA (*Direct Memory Access* ou Acesso Direto á Memória). Basicamente, ele faz é transferir informação de um endereço fonte para um endereço destino, de forma autônoma, sem intervenção do microcontrolador e sem a necessidade de se escrever código para efetuar essa operação a não se o necessário para a configuração do DMA. Apesar das transferências ocorrerem de forma autônoma, como o DMA acessa a mesma memória que o microcontrolador, durante uma operação de transferência o microcontrolador é paralisado, voltando ao seu estado normal ao término da operação, que demora dois ciclos de *clock*. Uma das principais aplicações do DMA nos MSP430 é na leitura do acumulador do conversor analógico-digital para a memória.

Pelo fato de os MSP430 utilizarem memória FLASH para o armazenamento de programas e dados, eles também contam com um controlador para manipular este tipo de memória. O controlador é um circuito com a finalidade de controlar o processo de apagamento e programação desse tipo de memória. Uma característica interessante é que todos os registradores do controlador são protegidos com senhas de escrita, o que impede que escritas equivocadas neles possam iniciar operações de apagamento ou programação indesejadas. Por isso, para realizar uma escrita, é necessário que o valor a ser escrito sempre possua o byte superior igual à senha de escrita, caso contrário ocasionará uma interrupção no microcontrolador.

Todos os modelos de MSP430 incluem um temporizador “cão de guarda” (*Watchdog*) com a finalidade de monitorar o aplicativo em execução no microcontrolador. O *watchdog* consiste basicamente de um contador de 16 bits que, ao final do período de contagem, que pode ser selecionado pelo usuário, pode provocar uma interrupção no microcontrolador, desviando o fluxo de execução do programa. É possível desligar esse módulo de forma a economizar energia.

Para a programação e depuração do aplicativo que será embarcado no MSP430, utilizada a interface JTAG (*Joint Test Action Group*), que consiste no meio de comunicação entre o equipamento externo de depuração e o hardware do microcontrolador. Para que efetivamente seja possível acompanhar e atuar sobre a execução do programa dentro do microcontrolador, é necessário que ele possua uma infra-estrutura interna preparada para isso, capaz de: iniciar e paralisar a execução do programa, criar pontos de parada (*breakpoints*) no programa em execução e realizar a leitura e escrita de qualquer posição de memória.

O modelo de MSP430 a ser utilizado no NanoSatC-BR é o MSP430F1611. Esse modelo trabalha a uma freqüência de 7.4 MHz em seu modo de operação normal. Ele possui 10 Kb de memória RAM e 50 Kb de memória FLASH, resultando em aproximadamente 64 Kb de memória compartilhada pelo mesmo espaço de endereçamento. É neste espaço que ficarão todos os dados do computador de bordo. A memória será dividida para acomodar diversos segmentos de código e dados, como o sistema operacional, o aplicativo de bordo e os dados obtidos pelas cargas úteis.

4.1.2. Carga útil

A maioria dos satélites possui objetivos específicos que são determinados pelos componentes das cargas úteis. A missão do NanoSatC-BR é obter dados sobre o campo magnético terrestre e sobre a precipitação de partículas altamente energizadas na superfície terrestre. Para isso, será utilizado o magnetômetro MAG566 desenvolvido pela Bartington Instruments e um detector de partículas desenvolvido por engenheiros do INPE.

O magnêtometro, quando acionado, produz três sinais analógicos correspondentes à intensidade do campo magnético terrestre nos eixos X, Y e Z. Esses sinais são enviados para as entradas analógicas do conversor

analógico-digital do microcontrolador e então convertidas em sinais digitais para posterior armazenamento em memória.

Já o detector de partículas, quando atingido por uma partícula altamente energizada, mede a energia depositada no seu material. Essa medida pode ser uma corrente gerada na colisão da partícula com o sensor do detector, ou apenas a diferença de potencial resultante da colisão. A medida então é recebida pelo microcontrolador através de uma entrada digital e armazenada na memória.

4.1.3. Subsistema de comunicação

O NanoSatC-BR utiliza um transceptor e uma antena para realizar a comunicação do satélite com a Terra.

- Transceptor

  Para modular a saída de informações e desmodular as informações recebidas, é utilizado a bordo do NanoSatC-BR um transceptor desenvolvido pela ISIS Company. Além de atuar como um modem (Modulator-Demodulator ou Modulador-Desmodulador), este dispositivos implementa o protocolo de comunicação AX.25 que padroniza o formato dos dados. Este componente é diretamente ligado ao computador de bordo utilizando o protocolo I^2C e à antena.

  Com isso, o computador de bordo é responsável pela comunicação serial com o transceptor.

- Antena

  O transceptor do NanoSatC-BR faz uso de uma antena dipolo desenvolvida pela ISIS Company. Antes de o satélite entrar em órbita, a antena está desativada, porém, uma vez o satélite em órbita, a antena deve ser ativada. O computador de bordo é responsável por esta ativação através de um sinal de controle.

Com isso, o computador de bordo é responsável pela ativação da antena logo após a estabilização do satélite em sua órbita assim como quando o satélite estiver em posição para enviar dados à Terra.

4.1.4. Subsistema de potência

No NanoSatC-BR, a obtenção, o armazenamento e a distribuição de energia é feita através de um subsistema de potência desenvolvido pela Clyde Space. A energia é gerada através de células solares e armazenada em baterias de Íon-Lítio. Esse subsistema possui um microcontrolador dedicado com a finalidade de monitorar as fontes de energia e diversos circuitos para medição de voltagem, medição de corrente e carga das baterias.

A interface de comunicação com o computador de bordo é realizada através do protocolo I^2C. O computador de bordo é responsável pelo envio de sinais para ativação de um determinado subsistema ou para obtenção de dados de "*housekeeping*".

CAPÍTULO 5

5.1. PLATAFORMA DE DESENVOLVIMENTO

Basicamente, para desenvolver um aplicativo é necessário um compilador compatível com a linguagem em que ele será escrito. Porém, pode-se agregar diversas outras ferramentas ao processo de desenvolvimento com a finalidade de tornar o processo de desenvolvimento mais robusto, como um depurador e um simulador, por exemplo.

Por isso, para a fase de programação e testes do aplicativo de bordo do NanoSatC-BR, é utilizado o compilador *CrossWorks for MSP430*, que é analisado a seguir.

O *CrossWorks for MSP430* é um compilador que faz parte do CrossStudio, que consiste em um conjunto de ferramentas desenvolvido pela Rowley Associates com suporte a grande parte de modelos de microcontroladores da família MSP430.

Voltado para o desenvolvimento na linguagem de programação C, o *CrossWorks for MSP430* é totalmente compatível com os padrões de programação ANSI e ISO, o que faz com que o compilador gere aplicativos com alta qualidade de código e desempenho para uso em microcontroladores da família MSP430. Além do compilador de C, o CrossStudio também conta com: suporte ao desenvolvimento na linguagem de baixo nível Assembly; um ligador; bibliotecas padrão de código da linguagem C; um simulador; interface com memórias FLASH; um depurador baseado na interface JTAG; e a IDE (*Integrated Development Environment* ou Ambiente Integrado de Desenvolvimento) propriamente dita, o CrossStudio, onde todas as funcionalidades anteriormente citadas são reunidas.

O CrossStudio tem como propósito, facilitar o desenvolvimento do aplicativo, englobando diversas ferramentas em uma única aplicação, como um editor de código, um gerenciador de projetos, um simulador, entre outras. Com

isso, pode-se desenvolver completamente um programa em uma única janela do computador, realizar simulações e enviá-lo diretamente para o MSP430 para a realização de testes e depuração do código.

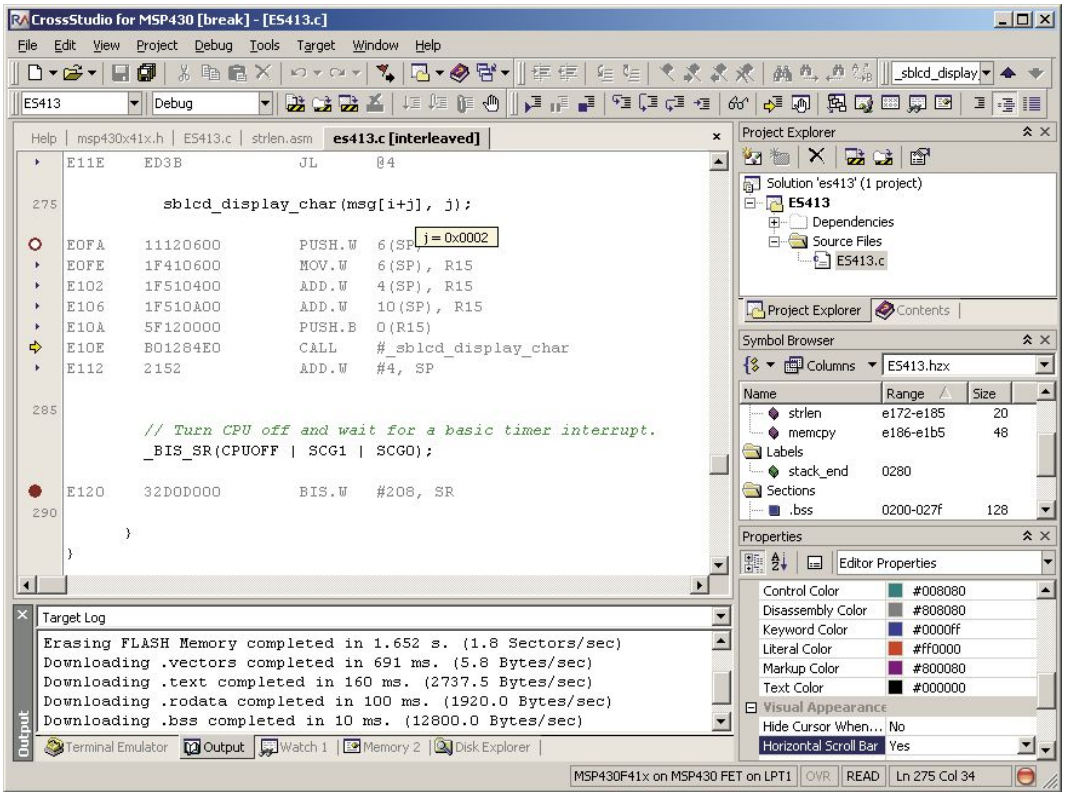


Figura 09: Tela principal do CrossStudio.[7]

Outra ferramenta do CrossStudio é o visualizador de símbolos (*Symbol Browser*). Nele é possível obter dados a respeito do tamanho da aplicação, sobre o tipo das variáveis e o tamanho de uma função. Essas informações são de extrema importância no desenvolvimento de sistemas embarcados, onde a memória disponível geralmente é muito limitada, tornando a localização de um byte, muitas vezes, vital.

[7] Disponível em <http://www.rowley.co.uk/>. Acesso em 15/05/2009.

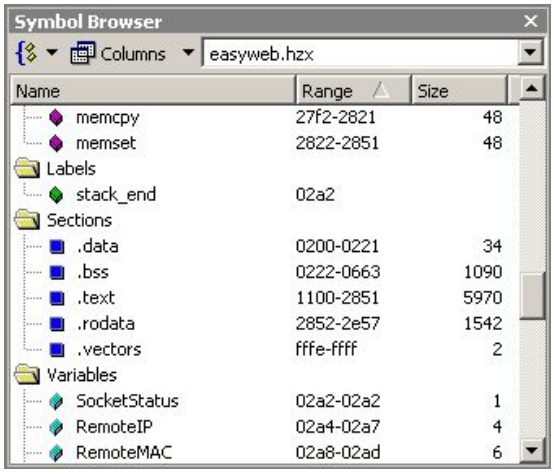


Figura 10: Visualizador de símbolos do CrossStudio.[8]

O CrossStudio gerencia os projetos através de um explorador de projetos (*Project Explorer*). Nele pode-se criar múltiplos projetos, cada um com seus arquivos e bibliotecas de código, havendo a possibilidade agrupá-los em uma única solução. O explorador de projetos também suporta os mesmo projetos com diferentes configurações, de forma que cada um seja testado e depurado com diferentes objetivos ou de diversas formas com um mesmo objetivo. Esta é a solução ideal para a fase de testes de um aplicativo, onde o mesmo pode se adaptar a diferentes configurações de hardware, por exemplo.

[8] Disponível em <http://www.rowley.co.uk/>. Acesso em 15/05/2009.

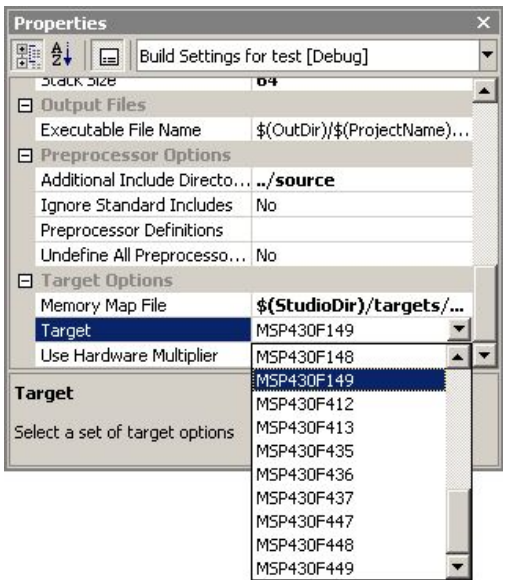


Figura 11: Diversas possibilidades de configurações de um projeto no CrossStudio.[9]

Uma das principais obrigações de um aplicativo espacial é que ele deve ser tolerante a falhas, e para que isso ocorra é necessário que haja um mecanismo de depuração eficiente. O CrossStudio possui um depurador com três itens que se destacam: suporte à inserção de *breakpoints* ao longo do programa*, Data Tips* e *Watch Window. Breakpoints* são utilizados para executar um programa passo-a-passo, com a finalidade de acompanhar a sua execução e seu fluxo de dados. *Data Tips* é uma janela exibida pelo CrossStudio para o detalhamento de variáveis locais, conteúdo de uma parte da memória e de expressões. *Watch Window* é um conjunto de janelas responsáveis por monitorar em tempo real uma variável, uma expressão, uma estrutura ou um ponteiro.

[9] Disponível em <http://www.rowley.co.uk/>. Acesso em 15/05/2009.

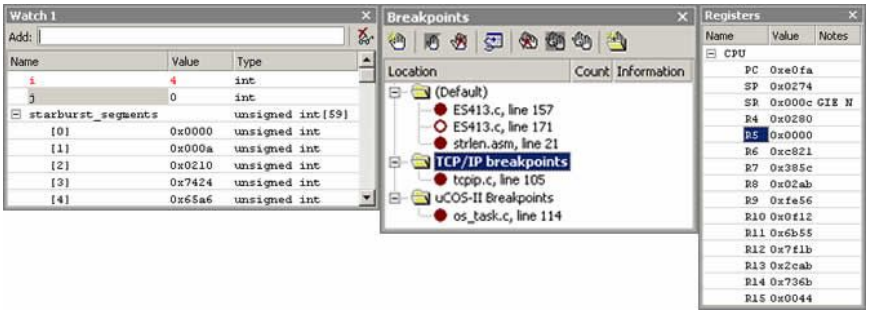


Figura 12: Exemplo de uma *Watch Window,* de *Breakpoints* e de uma *DataTips.*[10]

No CrossStudio, durante o processo de ligação, etapa que ocorre posterior a compilação, é possível interagir com o ligador. Com isso, é possível localizar o aplicativo na memória e agrupar pedaços de código com a finalidade de otimizar o aplicativo e o uso da memória.

Por fim, apesar de o CrossStudio ser voltado à programação na linguagem C, é possível desenvolver aplicativos em Assembly, isto é, em uma linguagem de programação com um nível mais baixo de abstração do hardware, com os mesmos benefícios da programação em C.

[10] Disponível em <http://www.rowley.co.uk/>. Acesso em 15/05/2009.

CAPÍTULO 6

6.1. ANÁLISE DO FLUXO DE DADOS DE BORDO

O objetivo deste capítulo é analisar o fluxo de dados entre os subsistemas do NanoSatC-BR, para então, através de diagramas, mostrar como os dados são processados pelo computado de bordo. A Figura 13, a seguir, descreve esses subsistemas em alto nível e a comunicação entre o satélite e a estação terrestre.

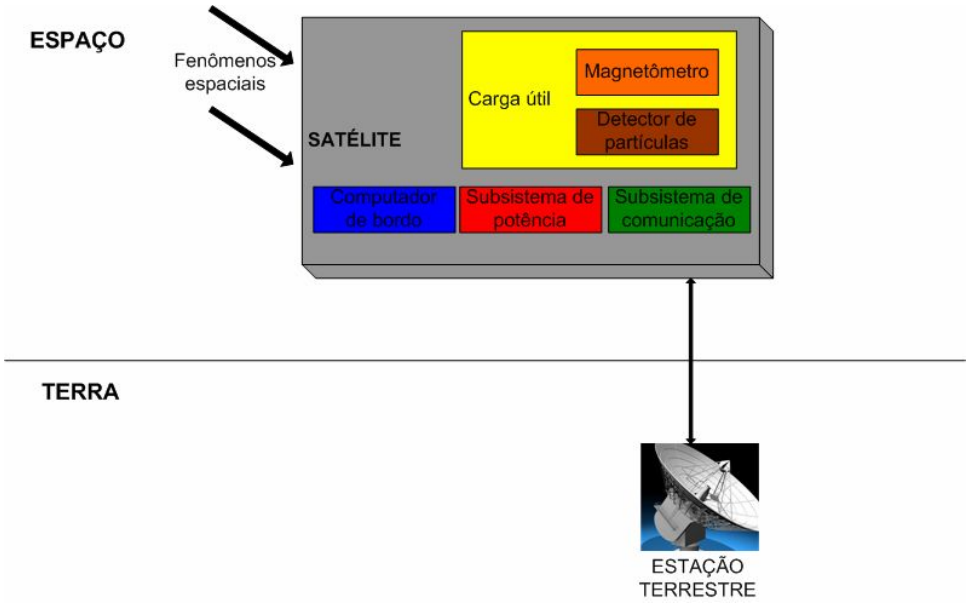


Figura 13: Distribuição dos subsistemas do satélite e a comunicação com a estação terrestre.

O computador de bordo é o principal subsistema, pois ele é o responsável pela gerência dos demais. Ele monitora as telemetrias obtidas, os telecomandos recebidos, os outros subsistemas e o armazenamento de dados. Com isso, ele pode ser considerado uma interface para o fluxo de dados entre a estação terrestre e os subsistemas que o compõe.

A comunicação dos subsistemas do NanoSatC-BR com o seu computador de bordo é realizada através de um canal linear que, utilizando o protocolo I^2C, oferece diversas vantagens, como o modo de operação mestre-

escravo, em que o computador de bordo pode controlar e se comunicar com todos os dispositivos através de um único canal, simplificando o fluxo de dados dentro do satélite.

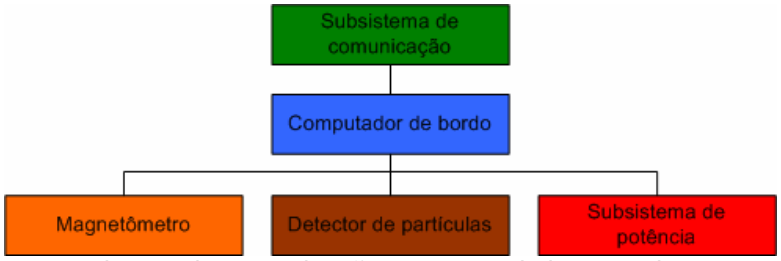


Figura 14: Arquitetura de comunicação entre os subsistemas do NanoSatC-BR.

6.1.1. Especificação dos requisitos do sistema

A seguir é realizada uma descrição simplificada dos requisitos funcionais do NanoSatC-BR, que serve de base para a análise do fluxo de dados dentro do satélite e para definição das limitações do sistema. Essa visão é essencial à compreensão dos tipos de interface com o hardware que o aplicativo deve ter e as restrições de desempenho associadas à missão, ao ambiente e às necessidades operacionais.

O computador de bordo tem como propósito gerenciar todo o processo de coleta e manipulação de dados dentro do satélite, receber um telecomando e enviar para estação terrestre dados de telemetria e do estado operacional do satélite (“*housekeeping”*) adquiridos nos demais subsistemas durante o intervalo de tempo entre duas visadas.

O processamento dos dados está divido em tarefas, onde cada subsistema do satélite possui as suas. Essas tarefas obedecem a uma fila de prioridade para serem executadas, atuando em cada um dos subsistemas de forma imediata ou temporizada conforme critérios de escalonamento estabelecidos. Os dados adquiridos dos subsistemas são armazenados para serem transmitidos à estação terrestre durante o período da próxima visada.

Dados de telemetria provenientes das cargas úteis são armazenados pelo computador de bordo. Também, dados de "*housekeeping"* do subsistema de potência e do computador de bordo são obtidos. Cada pacote de telemetria ou "*housekeeping"* possui um formato de armazenamento e um espaço de memória específico.

O computador de bordo deve prover o controle sobre os subsistemas existentes, como a ativação ou desativação de um subsistema, ajuste de data e carregamento de parâmetros. Além disso, ele deve ainda monitorar e atuar no seu próprio estado de funcionamento através de rotinas, como relato de eventos e tratamento de exceções.

Uma das principais funções do computador de bordo é receber um telecomando que sinaliza ao satélite que ele está visível pela estação terrestre e então, apto a enviar dados à Terra. Caso esse telecomando não seja recebido dentro de certo tempo, o envio de dados é iniciado. O envio desse telecomando ao satélite, assim como o envio de dados à Terra somente são realizados durante o período de visada.

A entidade externa Estação Terrestre envia o telecomando previamente definido ao satélite para ser executado de forma imediata, recebendo do mesmo, dados de telemetria e "*housekeeping"* armazenados.

Com isso, um diagrama de fluxo de dados em nível de contexto é esquematizado na figura 15. As entidades externas ao aplicativo de bordo produzem informações para serem usadas por ele e recebem informações dele. As setas rotuladas representam itens de dados compostos, ou seja, um item de dados que é, de fato, uma coleção de vários itens de dados.

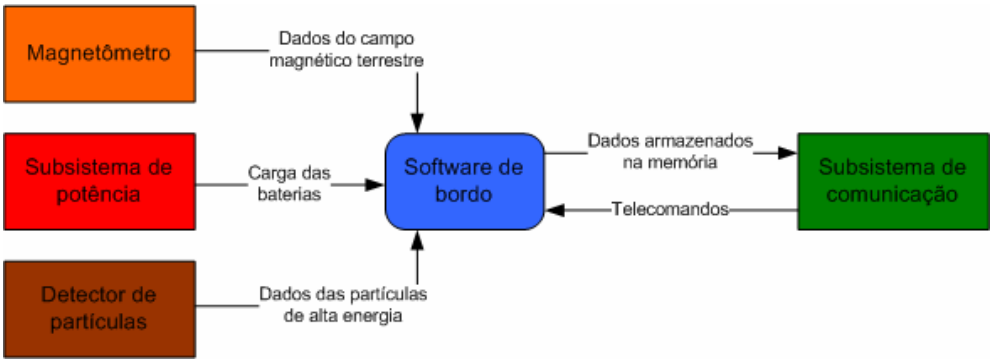


Figura 15: Diagrama de fluxo de dados do NanoSatC-BR em nível de contexto.

6.1.2. Análise estruturada

A partir da especificação dos requisitos do aplicativo, um modelo do fluxo de dados em nível de sistema é constituído, conforme mostra a figura 16.

Esse modelo obtido retrata que o aplicativo é mais voltado ao armazenamento de dados do que ao processamento, pois a maior parte dos processos coleta dados e os armazenam em memória. Deve-se ressaltar que no modelo desenvolvido, muitas informações não são apresentadas, como por exemplo, o tempo em que uma tarefa é executada e detalhes de hardware e eletrônica. Os processos são considerados processadores de dados e os fluxos somente transportam dados com valores, não são representadas informações sobre o fluxo de controle ou seqüência temporal, pois esse item é apenas uma referência para a implementação do aplicativo, que será abordada futuramente como proposta de pesquisa do bolsista.

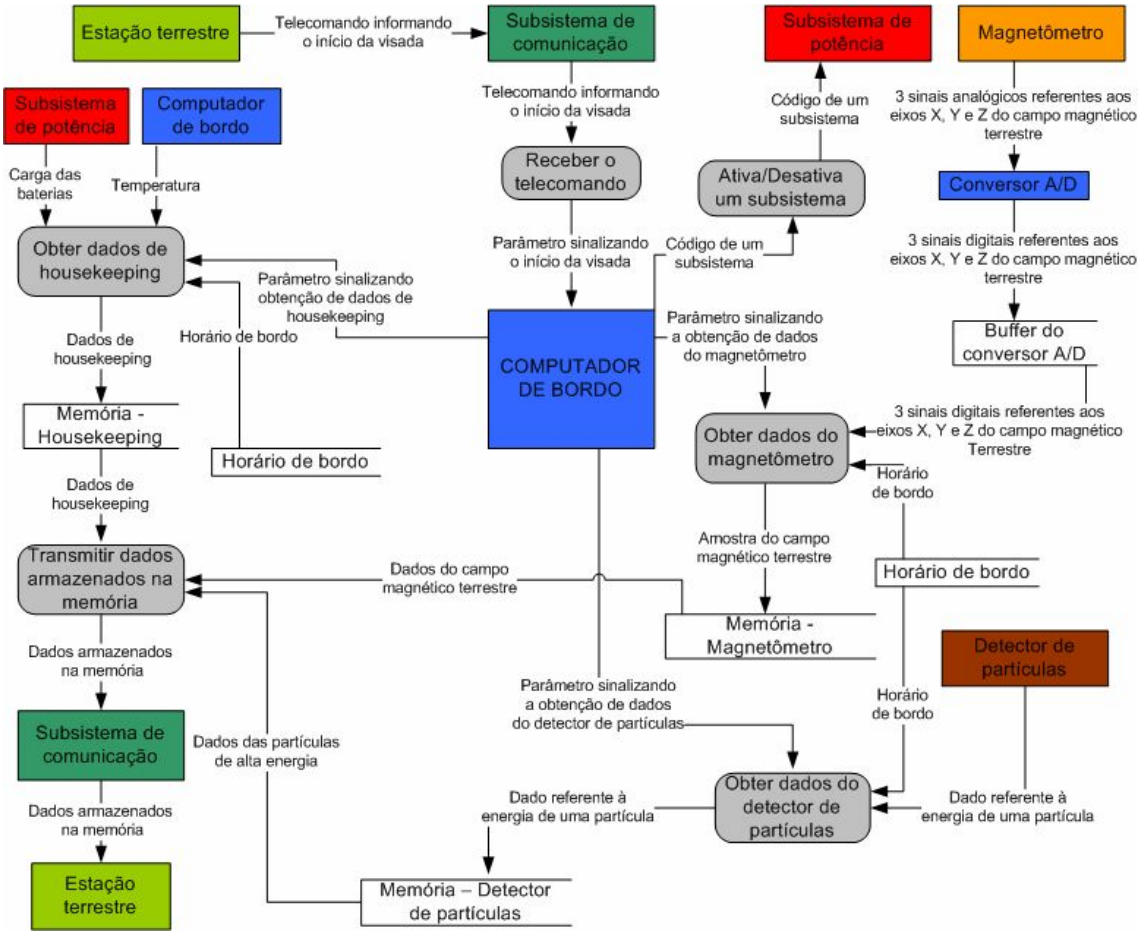


Figura 16: Diagrama de fluxo de dados do NanoSatC-BR em nível de sistema.

6.1.3. Especificação dos processos

Com o diagrama de fluxo de dados em nível de sistema estruturado, é realizada, a seguir, a especificação dos processos que compõe o diagrama com a finalidade de servir de guia para o projeto do aplicativo que implementará os processos.

- Receber o telecomando: aceita telecomandos como entrada de dados. O processo avalia os sinais recebidos para identificar o sinal referente à estação terrestre do NanoSatC-BR. A saída de dados é um sinal emitido ao computador de bordo sinalizando que o satélite está visível pela sua estação terrestre.

- Ativa/Desativa um subsistema: aceita códigos referente aos subsistemas do satélite como entrada de dados. O processo avalia os sinais recebidos e verifica o estado do subsistema. Caso o subsistema esteja desligado, ele é ligado, caso contrário, ele é desligado. A saída de dados é o mesmo código da entrada de dados mais um bit sinalizando a ativação ou desativação do subsistema em questão.
- Obter dados do magnetômetro: possui como entrada de dados três sinais digitais referentes aos eixos X, Y e Z do campo magnético terrestre e o horário de bordo. O processo reúne os dados recebidos como um único pacote de telemetria. A saída de dados consiste no pacote de telemetria formado que corresponde a uma amostra do campo magnético terrestre.
- Obter dados do detector de partículas: possui como entrada de dados um valor referente a uma partícula de alta energia e o horário de bordo. O processo reúne os dados recebidos como um único pacote de telemetria. A saída de dados consiste no pacote de telemetria formado correspondente a uma partícula de alta energia que colidiu com o satélite.
- Obter dados de *housekeeping*: possui como entrada de dados um dado referente à carga das baterias, um dado referente à temperatura interna do computador de bordo e o horário de bordo. O processo reúne os dados recebidos em um único pacote de telemetria. A saída de dados consiste no pacote de telemetria formado que corresponde a uma amostra da saúde do satélite.
- Transmitir dados armazenados na memória: possui como entrada de dados o conteúdo armazenado em memória, isso é, os pacotes de telemetria obtidos pelo magnetômetro, pelo detector de partículas e pelo processo de obtenção de dados de *housekeeping*. O processo reúne os dados recebidos para que eles sejam encapsulados no protocolo AX.25 pelo transceptor, protocolo esse designado para uso em rádios

amadores. A saída de dados consiste no conteúdo armazenado em memória para envio ao subsistema de comunicação.

CAPÍTULO 7

7.1. ESTRUTURAÇÃO DO APLICATIVO DE BORDO

Este capítulo tem como proposta a estruturação de um aplicativo de bordo compatível com as necessidades do NanoSatC-BR, descrevendo a sua organização, funções e tipos de dados.

O aplicativo é responsável por controlar o satélite através da sua execução no computador de bordo. Ele é organizado em três partes, a seqüência de inicialização, o aplicativo propriamente dito e a seqüência de reinicialização.

A seqüência de inicialização é executada no momento em que o satélite entra em órbita e tem como objetivo realizar a configuração do aplicativo principal para, em caso de sucesso, colocá-lo em execução.

O aplicativo é responsável por gerenciar o processamento no computador de bordo. Ele é organizado através de um sistema de tarefas oferecido pelo Salvo RTOS, um sistema operacional de tempo real desenvolvido pela Pumpkin Inc. e compatível com o microcontrolador em uso.

A seqüência de reinicialização é executada quando algum erro é detectado pelo aplicativo principal. Após a execução dos procedimentos que compõe essa parte do sistema, a seqüência de inicialização é requisitada para colocar o aplicativo de bordo em operação novamente.

Uma tarefa é uma seqüência de instruções a serem executadas para produzir uma ação, ou seja, é um pequeno programa dentro de um contexto. Quando uma tarefa é executada em um processador, ela pode obter todos os recursos do sistema para si própria, independente de quantas tarefas compõe o sistema.

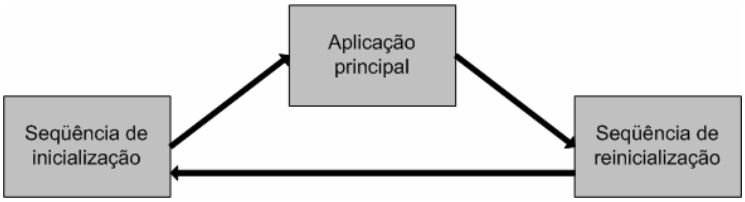


Figura 17: Partes do aplicativo de bordo.

7.1.1. Sequência de inicialização

A seqüência de inicialização é responsável por, de forma segura, configurar o aplicativo principal e colocá-lo em execução.

Primeiramente, o sistema operacional é configurado, criando as estruturas necessárias para que ele possa manipular tarefas e eventos. Nesse ponto, o sistema de tarefas ainda não está alocado.

Após a configuração do sistema operacional, diversas tarefas são alocadas e configuradas. Na configuração, é dado a cada tarefa um nível de prioridade de execução para que o sistema operacional escalone de forma eficiente todas elas sem que haja atrasos e/ou conflitos. Entre as tarefas alocadas estão as responsáveis por dar suporte a outras, como tarefas que fornecem o *clock* e o horário interno do sistema; as referente a cada subsistema do NanoSatC-BR; a responsável pela comunicação entre as outras tarefas e a que realiza a recuperação do sistema em caso de erro em algum procedimento e inicialização. Feita a alocação das tarefas, elas são inicializadas e habilitadas para execução. A execução ocorre partindo-se da tarefa de maior prioridade para as demais.

Esse contexto pode ser observado a seguir na Figura 18.

Figura 18: Seqüência de inicialização do aplicativo de bordo.

7.1.2. Aplicativo de bordo

O aplicativo de bordo é a peça central do computador de bordo do NanoSatC-BR. É ele que aloca tempo de processamento para todas as tarefas em execução, com a finalidade de proporcionar uma visão unificada do satélite, ao invés de cada subsistema possuir a sua unidade de processamento. Essa visão exige que o aplicativo tenha a capacidade de iniciar e parar uma determinada tarefa conforme especificado na configuração inicial do sistema. Por esses motivos, a seguir será descrito a estruturação do aplicativo de bordo.

As tarefas do aplicativo de bordo no formato de um diagrama são mostradas na figura 19. A tarefa “principal”, como o próprio nome mostra, é a mais importante do aplicativo, uma vez que é ela quem toma as decisões no sistema, como por exemplo, a realização de uma chamada para uma função em uma tarefa a ser executada.

As tarefas em blocos coloridos correspondem cada uma a um determinado subsistema do NanoSatC-BR, como as cargas úteis (megnetômetro com a tarefa “magnetometro” e o detector de partículas com a tarefa “dParticulas”), o subsistema de potência com a tarefa “potencia” e o de comunicação com a tarefa “comunicacao”.

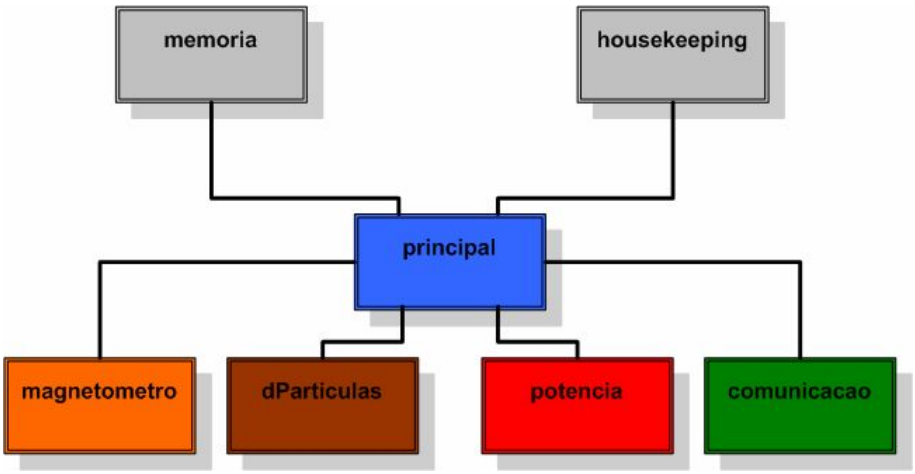


Figura 19: Estrutura do aplicativo de bordo.

As tarefas em blocos cinza são tarefas responsáveis pela gerência do sistema, ou seja, a tarefa “memoria” tem a função de manipular o espaço de endereçamento da memória do computador de bordo e a tarefa “*housekeeping*” tem a função de monitorar a saúde do satélite.

Pelo fato do aplicativo de bordo ser executado através do sistema operacional de tempo real *Salvo RTOS*, cada tarefa é mantida em um sistema de estados de execução. A figura 20 ilustra os diferentes estados em que uma tarefa pode se encontrar, e todas as transições possíveis. Uma tarefa *Pronta* é aquela que está pronta para ser executada, mas não é executada porque há tarefas de maior prioridade na fila de execução. Uma tarefa no estado *Executando* é aquela que possui com exclusividade os recursos do processador. O estado *Atrasada* faz referência às tarefas que estão suspensas esperando que o temporizador expire para que elas retornem ao estado *Pronta*. Uma tarefa está no estado *Parada* porque ela foi suspensa indefinidamente por algum motivo, e só irá voltar para o estado *Pronta* através de uma chamada do sistema operacional. O estado *Esperando* contém as tarefas estão suspensas e permanecerão assim até que o evento que ela está esperando ocorra. E por fim, uma tarefa no estado *Destruída* é aquela que não está inicializada.

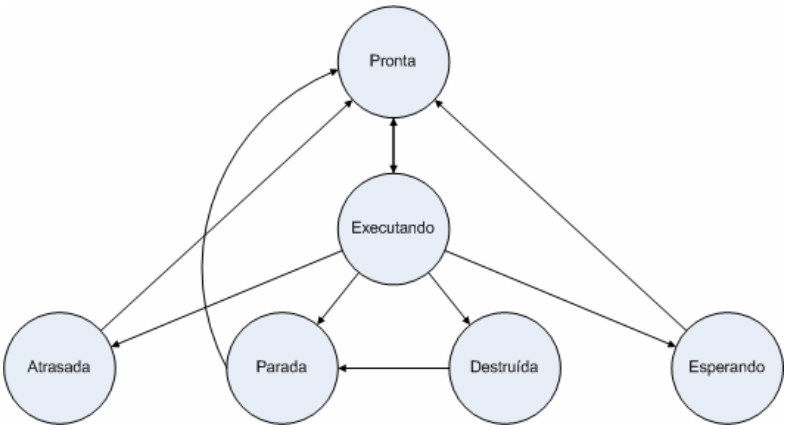


Figura 20: Diagrama de estados do *Salvo RTOS*.

A comunicação entre as tarefas é dada utilizando-se mensagens, ou seja, cada tarefa possui um identificador que pode ser um número, uma palavra, uma frase, uma função, uma estrutura ou um ponteiro. Para o NanoSatC-BR será utilizada uma estrutura de dados que conterá os seguintes dados: uma palavra identificando de que tarefa a mensagem provêm, como por exemplo, a palavra *PRI* identificando que a mensagem vem da tarefa principal do aplicativo; uma palavra com o nome da função que o remetente quer que o destinatário execute, como por exemplo, a palavra “ObtemDados” indicará que o remetente quer obter uma amostra do campo magnético terrestre se ela for enviada à tarefa responsável pelo magnetômetro; um ponteiro para o retorno da função chamada no destinatário e uma variável contendo o tamanho dos dados retornados.

Deve-se observar que os processos especificados no capítulo anterior (6.1.3) sofreram algumas modificações, como a junção dos processos *transmitir dados armazenados na memória* e *receber telecomando* em uma única tarefa chamada *comunicacao* para que cada subsistema possua somente uma tarefa, centralizando assim as suas funcionalidades. Porém, a especificação de cada processo é a mesma, não alterando a finalidade de cada um. Também, duas tarefas foram criadas: a tarefa *principal* foi criada para que ela controle o funcionamento do aplicativo de bordo, assumindo o papel do

computador de bordo em nível de aplicação; e a tarefa *memoria* foi criada para manipular o acesso e alocação de memória no computador de bordo.

Deve ser esclarecido que no capítulo anterior (6.1.3), durante a especificação dos processos foi descrito a entrada e saída de dados de cada processo (agora, em nível de aplicação já são tarefas), porém, de uma forma generalizada, e não os dados que cada função de um processo aceita como entrada e retorna como saída. Isso será especificado na continuidade desse trabalho, na fase de implementação.

A seguir, as tarefas *principal* e *memoria*, que não foram descritas na especificação de processos do capítulo anterior (6.1.3), são detalhadas:

- principal

  É a tarefa principal, que representa o computador de bordo em nível de aplicação. Ela é responsável por gerenciar o fluxo de dados dentro da aplicação através do sistema de mensagens descrito anteriormente.

  Pelo fato do aplicativo de bordo atuar como um aplicativo de tempo real, é necessário haver uma tarefa responsável pela sincronização de todas as outras. É a tarefa *principal* que atua como sincronizadora, ou seja, é ela quem mantém uma fila de execução de tarefas, fazendo com que o sistema se mantenha sincronizado, diminuindo a latência de execução.

  Quando ela recebe alguma mensagem, a tarefa analisa essa mensagem para descobrir de qual subsistema ela veio e para qual subsistema ela quer ir. Por exemplo, a tarefa *housekeeping* envia para a *principal* uma mensagem indicando que ela quer obter a carga atual das baterias e a temperatura do computador de bordo. A tarefa *principal* irá então criar uma mensagem para os subsistemas responsáveis pelos dados em questão indicando que ela quer tais dados, esperando assim uma resposta. Quando os dados requisitados retornarem, a tarefa

principal irá enviá-los para a tarefa *housekeeping*, que foi quem realmente os requisitou.

Além de gerenciar o fluxo de dados no aplicativo, a tarefa *principal* analisa os dados que ela recebe, para que, caso encontre algum valor crítico, adote um procedimento adequado para que o sistema se mantenha estável. Por exemplo, se ela estiver manipulando dados do subsistema de potência e receber uma amostra da carga das baterias abaixo do normal, irá entrar em modo de baixo consumo de energia.

- memoria

Tarefa que tem como responsabilidade realizar a alocação de memória para pacotes de telemetria ou de tarefas. Se o pacote for uma mensagem interna do satélite, ela deve ser sempre alocada, e, caso não havendo mais espaço disponível, a tarefa mais velha e de menor prioridade na memória é desalocada. Caso o pacote seja de telemetria, ele é criado normalmente se houver espaço disponível no seu segmento de memória, caso contrário o pacote mais antigo no segmento de memória em questão é apagado.

Essa abordagem será adotada por causa da grande limitação de memória e também pelo fato de que está se levando em conta que os pacotes de telemetria mais antigos não são os mais importantes para o NanoSatC-BR. Uma vez que o objetivo principal é estudar a região da Anomalia Magnética do Atlântico Sul (AMAS), e que a estação terrestre está localizada nessa região, admite-se que os últimos pacotes obtidos são os mais importantes, e por isso devem ser alocados.

A seguir, na figura 21, é exibido o formato de pacote enviado pela tarefa *principal* e recebido pela tarefa *memoria*. O campo *endereço* diz em qual segmento de memória ele deve ser armazenado, ou seja, no segmento de dados do magnetômetro ou do detector de partículas. O campo *tamanho* especifica o espaço de memória que deve ser alocado para os dados contidos no pacote. O campo *dado* contém as

informações que são armazenadas realmente em memória, ou seja, a data em que o dado (amostra) foi obtido e o dado propriamente dito.

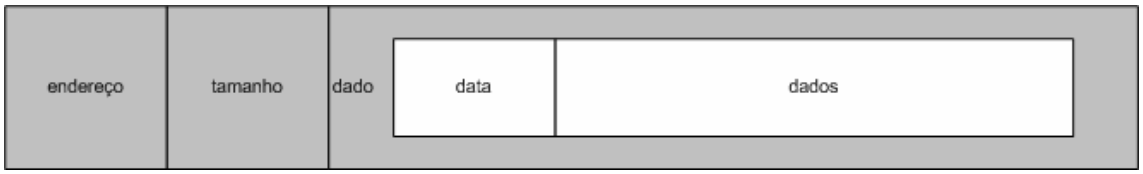


Figura 21: Formato de pacote recebido pela tarefa *memoria.*

Outra questão que não foi abordada no capítulo anterior (6.1.3) é leitura dos dados armazenados em memória pela tarefa *comunicacao*. Nela, a leitura é feita através da ordem crescente da data de obtenção das amostras e, à medida que os dados são transmitidos ao transceptor, eles são apagados da memória para dar lugar a novas amostras de dados que, se houver tempo disponível, serão transmitidos na mesma visada do satélite pela estação terrestre. Por causa desse comportamento, a informação não é transmitida à tarefa *comunicacao*, o que realmente é enviado é apenas um ponteiro apontando para o pacote armazenado em memória que será enviado ao transceptor, o que facilita o apagamento do mesmo.

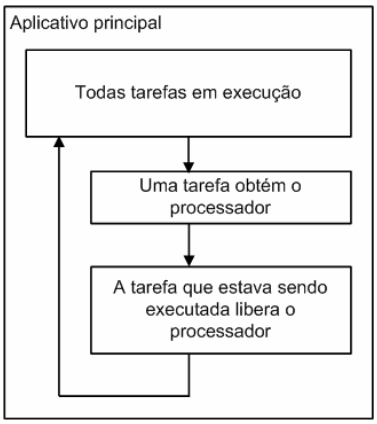


Figura 22: *Loop* do aplicativo de bordo.

A figura 22 exemplifica o funcionamento do aplicativo principal. Após a inicialização da aplicação, todas as tarefas são inicializadas e colocadas na fila de execução da tarefa *principal*. Assim que uma tarefa é seleciona, ela obtém o

processador e seus recursos, o liberando quando terminar os seus procedimentos ou quando ele sofrer uma interrupção.

7.1.3. Sequência de reinicialização

A seqüência de reinicialização é um modo de operação alternativo do satélite que é executado antes do sistema ser reiniciado.

Nessa parte do aplicativo de bordo, é executado um laço muito simples. As tarefas são finalizadas e o segmento de memória utilizado pelo aplicativo principal, que consiste nas tarefas e suas variáveis, é apagado. Após esses procedimentos, a seqüência de inicialização é chamada para que o aplicativo volte à sua execução normal.

CAPÍTULO 8

8.1. CONCLUSÃO

O Relatório descreve as atividades desenvolvidas pelo aluno no Projeto SOFTWARE DE BORDO PARA UM CUBESAT (NANOSATC-BR), no período de abril de 2009 a julho de 2009. Os resultados obtidos com esta pesquisa sintetizam os principais conceitos dos chamados CubeSats, suas dimensões e limitações, tendo como foco o Subsistema de Computador de Bordo.

Após uma descrição do Subsistema de Computador de Bordo e todos os assuntos com ele relacionados, foram mostrados os componentes e funcionalidades dos subsistemas que serão utilizados no satélite do Projeto NanoSatC-BR. Além do processo e dos passos a seguir para o desenvolvimento de um Aplicativo de Bordo para um satélite dessa classe, o bolsista fez uma análise de solução para o aplicativo de bordo do NanosatC-BR.

Com estas atividades o bolsista desenvolveu habilidades de pesquisa, conhecimento e maior entendimento dos conceitos técnicos aplicados na área espacial e experiência com a parte prática de Projeto. Os estudos foram importantes para o crescimento profissional e pessoal do bolsista, tanto no aprimoramento técnico, como no seu desenvolvimento em áreas de formação pessoal, como liberdades pessoais, auto-estima, autoconfiança, aprimorando suas habilidades de autodidatismo, liderança, iniciativa e criatividade.

8.2. TRABALHOS FUTUROS

O bolsista pretende continuar atuando na pesquisa sobre Aplicativos de Bordo de Cubesats, entretanto, dando maior ênfase para a aplicação dos conceitos e tecnologias ao Projeto NanosatC-BR.

Soluções otimizadas são efetivamente possíveis quando forem definidas outras especificações da Missão NanoSatC-BR, como: tipo de órbita, tempo de visibilidade do satélite pela estação terrestre, velocidade média de transferência de dados pelo subsistema de comunicação, entre outras.

Definindo a maioria dos requisitos, torna-se possível o desenvolvimento de um aplicativo de bordo compatível com a Missão NanoSatC-BR. Para isso, é necessária a realização de testes de verificação e validação no aplicativo para sua qualificação e utilização.

REFERÊNCIAS BIBLIOGRÁFICAS

CENTRO REGIONAL DE PESQUISAS ESPACIAIS – CRS/INPE. Projeto Básico – Missão NanoSatC-BR – Clima Eespacial, “Versão Um”. Santa Maria – RS, 2008.

DE SOUZA, P. N. Curso Introdutório de Tecnologia de Satélites. Instituto Nacional de Pesquisas Espaciais – INPE. São José dos Campos – SP, 2007.

Mattiello-Francisco, M. F. Sistemas Computacionais em Aplicações Espaciais. INPE-9604-PUD/125. Fev. 2003.

DE SOUZA, P. B. Melhoria do Software Embarcado em Satélites do INPE: proposta para um passo a mais. Dissertação (Mestrado em Computação Aplicada), Instituto Nacional de Pesquisas Espaciais – INPE, São José dos Campos, SP, 2007.

PRESSMAN, R. S. Engenharia de Software. São Paulo, SP. Makron Books do Brasil, 1995.

PEREIRA, F. Microcontroladores MSP430: Teoria e Prática. São Paulo, SP. Editora Érica, 2005.

Rowley Associates Online Documentation: CrossWorks for MSP430. Disponível em: < http://www.rowleydownload.co.uk/documentation/msp430_2_0/>. Acesso em: Abr. 2009.

Salvo 4 User Manual. Disponível em: < http://www.pumpkininc.com/content/doc/manual/SalvoUserManual.pdf/>. Acesso em: Jun. 2009.

AALBORG UNIVERSITY. Documentation of AAU-CubeSat On Board Computer Software. Allborg, Dinamarca, Nov. 2002.

DABROWSKI, M. J. The Design of a Software System for a Small Satellite. Dissertação (Mestrado em Engenharia Elétrica), Universidade de Illinois, Urbana, Illinois, Estados Unidos, 2003.

ATIVIDADES COMPLEMENTARES – PARTICIPAÇÃO E APRESENTAÇÃO DE TRABALHOS

1. COSTA, L. L.; SCHUCH, N. J. ; SOUZA, P. N.; DURÃO, O. S. C.; TRIVEDI, N. B.; MENDES JUNIOR, O. ; LOPES, R. V. F.; FONSECA, I. M.; SOUSA, F. L.; PALEROSI, A. C.; ROZENFELD, P.; GOMES, N.; MICHELS, A.; PINHEIRO, D. K. ; COSTA, R. L.; ZOLAR, R. B.; ANTUNES, C. E. ; SIQUEIRA, J. ; SAVIAN, F. S.; FAGUNDES, I. F.; JASKULSKI, T.; NICOLINI, L. F.; **TAMBARA, L. A.**; FELIPETTO, C. M.; SANTOS, R. C.; BURGER, E. E. . **NANOSATC-BR - SPACE WEATHER MISSION**. In: 11th International Congress of the Brazilian Geophysical Society, 2009, Salvador, BA. 11th International Congress of the Brazilian Geophysical Society, 2009.

2. COSTA, R. L.; SOUZA, P. N.; SCHUCH, N. J. ; DURÃO, O. S. C.; COSTA, L. L.; ZOLAR, R. B.; **TAMBARA, L. A.**. **NANOSATC-BR SUBSYSTEMS AND PAYLOAD**. In: 11th International Congress of the Brazilian Geophysical Society, 2009, Salvador, BA. 11th International Congress of the Brazilian Geophysical Society, 2009.